Package ‘spotifyr’

July 14, 2019

Title R Wrapper for the 'Spotify' Web API
Version 2.1.1
Description An R wrapper for pulling data from the 'Spotify' Web API
<http://developer.spotify.com/web-api> in bulk. It allows you to enter
an artist's name and retrieve their entire discography in seconds, along
You can also pull song and playlist information for a given 'Spotify' user.

Depends R (>= 3.3.0)
Imports dplyr, purrr, tidyr, httr, lubridate, jsonlite, readr, rvest,
stringr, tibble, genius
License MIT + file LICENSE
Encoding UTF-8
LazyData true
RoxygenNote 6.1.1
Suggests testthat
URL http://github.com/charlie86/spotifyr
BugReports http://github.com/charlie86/spotifyr/issues
NeedsCompilation no
Author Charlie Thompson [aut, cre],
Josiah Parry [aut],
Donal Phipps [aut],
Tom Wolff [aut]
Maintainer Charlie Thompson <chuck@rcharlie.com>
Repository CRAN
Date/Publication 2019-07-13 23:00:03 UTC

R topics documented:

    add_tracks_to_playlist ........................................ 3
    change_playlist_details ......................................... 4
R topics documented:

check_me_following ............................................ 4
check_users_following ........................................... 5
create_playlist .................................................. 6
dedupe_album_names ............................................ 6
follow_artists_or_users ....................................... 7
follow_playlist .................................................. 8
get_album ........................................................ 8
get_albums ...................................................... 9
get_album_data .................................................. 10
get_album_tracks .............................................. 10
get_artist ......................................................... 11
get_artists ....................................................... 12
get_artist_albums ............................................... 12
get_artist_audio_features .................................... 13
get_artist_top_tracks .......................................... 14
get_category ..................................................... 15
get_category_playlists ......................................... 16
get_discography ................................................. 17
get_featured_playlists ......................................... 17
get_genre_artists ............................................... 19
get_label_artists ............................................... 20
get_my_currently_playing .................................... 21
get_my_current_playback ..................................... 21
get_my_devices .................................................. 22
get_my_followed_artists ....................................... 23
get_my_playlists ............................................... 23
get_my_profile .................................................. 24
get_my_recently_played ....................................... 25
get_my_top_artists_or_tracks ................................ 26
get_new_releases ............................................... 27
get_playlist ..................................................... 28
get_playlist_audio_features ................................ 29
get_playlist_cover_image .................................... 29
get_playlist_tracks ............................................ 30
get_recommendations ............................................ 31
get_related_artists ............................................. 36
get_spotify_access_token .................................... 36
get_spotify_authorization_code ............................. 37
get_track ........................................................ 38
get.tracks ....................................................... 38
get_track_audio_analysis .................................... 39
get_track_audio_features .................................... 40
get_user_audio_features ...................................... 40
get_user_playlists ............................................. 41
get_user_profile ................................................ 42
is_uri ........................................................... 42
pause_my_playback ............................................. 43
pitch_class_lookup .............................................. 43
Add one or more tracks to a user's playlist.

Description

Add one or more tracks to a user’s playlist.

Usage

```r
add_tracks_to_playlist(playlist_id, uris, position = NULL,
authorization = get_spotify_authorization_code())
```

Arguments

- **playlist_id**: Required. The Spotify ID for the playlist.
- **uris**: Optional. A character vector of Spotify track URIs to add. For example uris = "spotify:track:4iV5W9uYEd4VaT9Ax7bRh", "spotify:track:1301WleyT98MSxVHPZCA6MyA". A maximum of 100 tracks can be added in one request.
- **position**: Optional. Integer indicating the position to insert the tracks, a zero-based index. For example, to insert the tracks in the first position: `position = 0`; to insert the tracks in the third position: `position = 2`. If omitted, the tracks will be appended to the playlist. Tracks are added in the order they are listed in the query string or request body.
- **authorization**: Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`. The access token must have been issued on behalf of the current user. Adding tracks to the current user’s public playlists requires authorization of the playlist-modify-public scope; adding tracks to the current user's private playlist (including collaborative playlists) requires the playlist-modify-private scope. See Using Scopes.
change_playlist_details

Change a playlist’s name and public/private state. (The user must, of course, own the playlist.)

Description

Change a playlist’s name and public/private state. (The user must, of course, own the playlist.)

Usage

change_playlist_details(playlist_id, name = null, public = null, collaborative = null, description = null, authorization = get_spotify_authorization_code())

Arguments

playlist_id Required. The Spotify ID for the playlist.
name Optional String containing the name for the new playlist, for example "Your Coolest Playlist". This name does not need to be unique; a user may have several playlists with the same name.
public Optional. Boolean. If TRUE the playlist will be public. If FALSE it will be private.
collaborative Optional. Boolean. If TRUE the playlist will become collaborative and other users will be able to modify the playlist in their Spotify client. Note: you can only set collaborative to TRUE on non-public playlists.
description Optional. String containing the playlist description as displayed in Spotify Clients and in the Web API.
authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to spotipy::get_spotify_authorization_code(). The access token must have been issued on behalf of the current user. Changing a public playlist for a user requires authorization of the playlist-modify-public scope; changing a private playlist requires the playlist-modify-private scope. See Using Scopes.

check_me_following

Check if Current User Follows Artists or Users

Description

Check if Current User Follows Artists or Users

Usage

check_me_following(type, ids, authorization = get_spotify_authorization_code())
check_users_following

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Required. String of the ID type: either &quot;artist&quot; or &quot;user&quot;.</td>
</tr>
<tr>
<td>ids</td>
<td>Required. A character vector of the artist or the user Spotify IDs to check. For example: <code>ids = c(&quot;74ASZWbe41XaubB36ztrGx&quot;, &quot;08td7MxkoHQkXnWAYD8d6Q&quot;)</code>. A maximum of 50 IDs can be sent in one request.</td>
</tr>
</tbody>
</table>
| authorization | Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`.

Following a playlist can be done publicly or privately. Checking if a user publicly follows a playlist doesn't require any scopes; if the user is publicly following the playlist, this endpoint returns `true`. Checking if the user is privately following a playlist is only possible for the current user when that user has granted access to the `playlist-read-private` scope. See Using Scopes.
create_playlist

Create a playlist for a Spotify user. (The playlist will be empty until you add tracks.)

Description

Create a playlist for a Spotify user. (The playlist will be empty until you add tracks.)

Usage

create_playlist(user_id, name, public = TRUE, collaborative = FALSE, description = NULL, authorization = get_spotify_authorization_code())

Arguments

user_id  Required. The user's Spotify user ID.

name  Required. String containing the name for the new playlist, for example "Your Coolest Playlist". This name does not need to be unique; a user may have several playlists with the same name.

public  Optional. Boolean. Defaults to TRUE. If TRUE the playlist will be public. If FALSE it will be private. To be able to create private playlists, the user must have granted the playlist-modify-private scope.

collaborative  Optional. Boolean. Defaults to FALSE. If TRUE the playlist will be collaborative. Note that to create a collaborative playlist you must also set public to FALSE. To create collaborative playlists you must have granted playlist-modify-public and playlist-modify-private scopes.

description  Optional. String containing the playlist description as displayed in Spotify Clients and in the Web API.

authorization  Required. A valid access token from the Spotify Accounts service. See the Web API Authorization Guide for more details. Defaults to spotifyr::get_spotify_authorization_code(). The access token must have been issued on behalf of the current user. Creating a public playlist for a user requires authorization of the playlist-modify-public scope; creating a private playlist requires the playlist-modify-private scope. See Using Scopes.

dedupe_album_names

Remove duplicate album names

Description

Use fuzzy matching to remove duplicate album names (including reissues, remasters, etc)
**Follow Artists or Users**

**Usage**

```r
dedupe_album_names(df, album_name_col = "album_name",
                   album_release_year_col = "album_release_year")
```

**Arguments**

- `df` Dataframe with album name
- `album_name_col` String of field name containing album names
- `album_release_year_col` String of field name containing album release year

---

**Follow Artists or Users**

*Add the current user as a follower of one or more artists or other Spotify users.*

**Description**

Add the current user as a follower of one or more artists or other Spotify users.

**Usage**

```r
follow_artists_or_users(type, ids,
                        authorization = get_spotify_authorization_code())
```

**Arguments**

- `type` Required. The ID type: either "artist" or "user".
- `ids` Optional. A character vector of the artist or the user Spotify IDs. For example: `ids = c("74ASZwbe41XaubB36ztrGx", "08td7MxkoHQkXnWAYD8d6Q")`. A maximum of 50 IDs can be sent in one request.
- `authorization` Required. A valid access token from the Spotify Accounts service. See the [Web API Authorization Guide](https://developer.spotify.com/documentation/general/guides/authorization-guide/) for more details. Defaults to `spotifyr::get_spotify_authorization_code()`. The access token must have been issued on behalf of the current user. Modifying the list of artists or users the current user follows requires authorization of the user-follow-modify scope. See [Using Scopes](https://developer.spotify.com/documentation/general/guides/using-scopes/).
follow_playlist

Add the current user as a follower of a playlist.

Description

Add the current user as a follower of a playlist.

Usage

follow_playlist(playlist_id, public = FALSE, authorization = get_spotify_authorization_code())

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>playlist_id</td>
<td>Required. The Spotify ID of the playlist. Any playlist can be followed, regardless of its public/private status, as long as you know its playlist ID.</td>
</tr>
<tr>
<td>public</td>
<td>Optional. Defaults to TRUE. If TRUE the playlist will be included in the user’s public playlists, if FALSE it will remain private. To be able to follow playlists privately, the user must have granted the playlist-modify-private scope.</td>
</tr>
<tr>
<td>authorization</td>
<td>Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to spotifyr::get_spotify_authorization_code(). The access token must have been issued on behalf of the current user. Following a publicly followed playlist for a user requires authorization of the playlist-modify-public scope; following a privately followed playlist requires the playlist-modify-private scope. See See Using Scopes. Note that the scopes you provide relate only to whether the current user is following the playlist publicly or privately (i.e. showing others what they are following), not whether the playlist itself is public or private.</td>
</tr>
</tbody>
</table>

get_album

Get Spotify catalog information for a single album.

Description

Get Spotify catalog information for a single album.

Usage

get_album(id, market = NULL, authorization = get_spotify_access_token())
**get_albums**

**Arguments**

- **id**: The Spotify ID for the album.
- **market**: Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking
- **authorization**: Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`

**Value**

Returns a data frame of results containing album data. See the official documentation for more information.

---

**get_albums**

Get Spotify catalog information for multiple albums identified by their Spotify IDs.

**Description**

Get Spotify catalog information for multiple albums identified by their Spotify IDs.

**Usage**

```r
get_albums(ids, market = NULL,
           authorization = get_spotify_access_token(),
           include_meta_info = FALSE)
```

**Arguments**

- **ids**: Required. A character vector of the Spotify IDs for the albums. Maximum: 20 IDs.
- **market**: Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking
- **authorization**: Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`
- **include_meta_info**: Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE

**Value**

get_album_data

Retrieve artist discography with song lyrics and audio info

Description

Retrieve the entire discography of an artist with the lyrics of each song and the associated audio information. Returns the song data as a nested tibble. This way we can easily see each album, artist, and song title before expanding our data.

Usage

```r
get_album_data(artist, albums = character(),
authorization = get_spotify_access_token())
```

Arguments

- `artist`: The quoted name of the artist. Spelling matters, capitalization does not.
- `albums`: A character vector of album names. Spelling matters, capitalization does not.
- `authorization`: Authorization token for Spotify web API. Defaults to `get_spotify_access_token()`.

Examples

```r
# Not run:
get_album_data("Wild child", "Expectations")

# End(Not run)
```

get_album_tracks

Get Spotify catalog information about an album’s tracks. Optional parameters can be used to limit the number of tracks returned.

Description

Get Spotify catalog information about an album’s tracks. Optional parameters can be used to limit the number of tracks returned.

Usage

```r
get_album_tracks(id, limit = 20, offset = 0, market = NULL,
authorization = get_spotify_access_token(),
include_meta_info = FALSE)
```
get_artist

Arguments

id
  Required. The Spotify ID for the album.
limit
  Optional. Maximum number of results to return.
  Default: 20
  Minimum: 1
  Maximum: 50
  Note: The limit is applied within each type, not on the total response.
  For example, if the limit value is 3 and the type is c("artist", "album"), the
  response contains 3 artists and 3 albums.
offset
  Optional. The index of the first album to return.
  Default: 0 (the first album).
  Maximum offset (including limit): 10,000.
  Use with limit to get the next page of albums.
market
  Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this
  parameter if you want to apply Track Relinking.
authorization
  Required. A valid access token from the Spotify Accounts service. See the Web
  API authorization Guide for more details. Defaults to spotifyr::get_spotify_access_token().
include_meta_info
  Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

Value

Returns a data frame of results containing album data. See the official API documentation for more information.

get_artist

Get Spotify catalog information for a single artist identified by their unique Spotify ID.

Description

Get Spotify catalog information for a single artist identified by their unique Spotify ID.

Usage

get_artist(id, authorization = get_spotify_access_token())

Arguments

id
  The Spotify ID for the artist.
authorization
  Required. A valid access token from the Spotify Accounts service. See the Web
  API authorization Guide for more details. Defaults to spotifyr::get_spotify_access_token()
get_artist_albums

**Value**


---

get_artists

*Get Spotify catalog information for multiple artists identified by their Spotify IDs.*

**Description**

Get Spotify catalog information for multiple artists identified by their Spotify IDs.

**Usage**

```r
get_artists(ids, authorization = get_spotify_access_token(),
            include_meta_info = FALSE)
```

**Arguments**

- `ids` Required. A character vector of the Spotify IDs for the artists. Maximum: 50 IDs.
- `authorization` Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to `spotifyr::get_spotify_access_token()`.
- `include_meta_info` Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to `FALSE`.

**Value**


---

get_artist_albums

*Get Spotify catalog information for multiple artists identified by their Spotify IDs.*

**Description**

Get Spotify catalog information for multiple artists identified by their Spotify IDs.

**Usage**

```r
get_artist_albums(id, include_groups = c("album", "single", "appears_on", "compilation"), market = NULL, limit = 20, offset = 0,
                  authorization = get_spotify_access_token(),
                  include_meta_info = FALSE)
```
get_artist_audio_features

Arguments

- **id**: The Spotify ID for the artist.
- **include_groups**: Optional. A character vector of keywords that will be used to filter the response. If not supplied, all album types will be returned. Valid values are:
  - "album"
  - "single"
  - "appears_on"
  - "compilation"
  For example: `include_groups = c("album", "single")`
- **market**: Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". Supply this parameter to limit the response to one particular geographical market. For example, for albums available in Sweden: `market = "SE"`. If not given, results will be returned for all markets and you are likely to get duplicate results per album, one for each market in which the album is available!
- **limit**: Optional. Maximum number of results to return.
  - Default: 20
  - Minimum: 1
  - Maximum: 50
- **offset**: Optional. The index of the first album to return.
  - Default: 0 (i.e., the first album).
  - Use with limit to get the next set of albums.
- **authorization**: Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to `spotifyr::get_spotify_access_token()`.
- **include_meta_info**: Optional. Boolean indicating whether to include full result, with meta information such as "total" and "limit". Defaults to FALSE.

Value


Description

Get audio feature information for all or part of an artists’ discography.
Usage

```r
get_artist_top_tracks(id, market = "US",
                      authorization = get_spotify_access_token(),
                      include_meta_info = FALSE)
```

Description

Get Spotify catalog information about an artist’s top tracks by country.
get_category

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The Spotify ID for the artist.</td>
</tr>
<tr>
<td>market</td>
<td>Required. An ISO 3166-1 alpha-2 country code or the string &quot;from_token&quot;.</td>
</tr>
<tr>
<td>authorization</td>
<td>Required. A valid access token from the Spotify Accounts service. See the</td>
</tr>
<tr>
<td></td>
<td>Web API authorization guide for more details. Defaults to spotifyr::get_spotify_access_token()</td>
</tr>
<tr>
<td>include_meta_info</td>
<td>Optional. Boolean indicating whether to include full result, with meta information such as &quot;total&quot;, and &quot;limit&quot;. Defaults to FALSE.</td>
</tr>
</tbody>
</table>

Value


get_category

Get a single category used to tag items in Spotify (on, for example, the Spotify player's "Browse" tab).

Description

Get a single category used to tag items in Spotify (on, for example, the Spotify player's "Browse" tab).

Usage

get_category(category_id, country = NULL, locale = NULL, authorization = get_spotify_access_token())

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>category_id</td>
<td>Required. The Spotify ID for the category.</td>
</tr>
<tr>
<td>country</td>
<td>Optional. A country: an ISO 3166-1 alpha-2 country code. Provide this parameter to ensure that the category exists for a particular country.</td>
</tr>
<tr>
<td>locale</td>
<td>Optional. The desired language, consisting of an ISO 639-1 language code and an ISO 3166-1 alpha-2 country code, joined by an underscore. For example: es_mx, meaning &quot;Spanish (Mexico)&quot;. Provide this parameter if you want the category strings returned in a particular language. Note that, if locale is not supplied, or if the specified language is not available, the category strings returned will be in the Spotify default language (American English).</td>
</tr>
<tr>
<td>authorization</td>
<td>Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to spotifyr::get_spotify_access_token()</td>
</tr>
</tbody>
</table>


get_category_playlists

Get a list of Spotify playlists tagged with a particular category.

Description

Get a list of Spotify playlists tagged with a particular category.

Usage

get_category_playlists(category_id, country = NULL, limit = 20,
offset = 0, authorization = get_spotify_access_token(),
include_meta_info = FALSE)

Arguments

category_id  Required. The Spotify ID for the category.
Maximum: 50.
offset  Optional. The index of the first item to return. Default: 0 (the first object). Use
with limit to get the next set of items.
authorization  Required. A valid access token from the Spotify Accounts service. See the Web
API authorization guide for more details. Defaults to spotyr::get_spotify_access_token()
include_meta_info  Optional. Boolean indicating whether to include full result, with meta informa-
tion such as "total", and "limit". Defaults to FALSE.

Value

Returns a data frame of results containing category playlists. See https://developer.spotify.com/
documentation/web-api/reference/users-profile/get-current-users-profile/ for more
information.

Examples

## Not run:
## Get Brazilian party playlists
get_category_playlists('party', country = 'BR')

## End(Not run)
get_discography

Retrieves the discography of an artist with the lyrics of each song and the associated audio information. Returns the song data as a nested tibble. This way we can easily see each album, artist, and song title before expanding the data.

Usage

```r
get_discography(artist, authorization = get_spotify_access_token())
```

Arguments

- `artist`: The quoted name of the artist. Spelling matters, capitalization does not.
- `authorization`: Authorization token for Spotify web API. Defaults to `get_spotify_access_token()`.

Examples

```r
CC not runZ
rex_orange <- get_discography("Rex Orange County")
unnest(rex_orange, data)
CC endHnot runI
```

get_featured_playlists

A list of Spotify featured playlists (shown, for example, on a Spotify player’s ‘Browse’ tab).

Description

Get a list of Spotify featured playlists (shown, for example, on a Spotify player’s ‘Browse’ tab).

Usage

```r
get_featured_playlists(locale = NULL, country = NULL,
                       timestamp = NULL, limit = 20, offset = 0,
                       authorization = get_spotify_access_token(),
                       include_meta_info = FALSE)
```
get_featured_playlists

Arguments

locale Optional. The desired language, consisting of an ISO 639-1 language code and an ISO 3166-1 alpha-2 country code, joined by an underscore. For example: es_MX, meaning "Spanish (Mexico)". Provide this parameter if you want the category strings returned in a particular language. Note that, if locale is not supplied, or if the specified language is not available, the category strings returned will be in the Spotify default language (American English). The locale parameter, combined with the country parameter, may give odd results if not carefully matched. For example country=SE&locale=de_DE will return a list of categories relevant to Sweden but as German language strings.

country Optional. A country: an ISO 3166-1 alpha-2 country code. Provide this parameter if you want the list of returned items to be relevant to a particular country. If omitted, the returned items will be relevant to all countries.

timestamp Optional. A timestamp in ISO 8601 format: yyyy-MM-ddTHH:mm:ss. Use this parameter to specify the user’s local time to get results tailored for that specific date and time in the day. If not provided, the response defaults to the current UTC time. Example: “2014-10-23T09:00:00” for a user whose local time is 9AM. If there were no featured playlists (or there is no data) at the specified time, the response will revert to the current UTC time.


offset Optional. The index of the first item to return. Default: 0 (the first object). Use with limit to get the next set of items.

authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to $spotifyr::get_spotify_access_token()$

include_meta_info Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

Value


Examples

```r
## Not run:
## Get new Swedish music
get_featured_playlists(country = 'SE')

## End(Not run)
```
get_genre_artists  

Get Spotify Catalog information about artists belonging to a given genre.

Usage

get_genre_artists(genre = character(), market = NULL, limit = 20,
offset = 0, authorization = get_spotify_access_token())

Arguments

genre  
Required.  
String of genre name to search for.  
For example: genre = "wonky".

market  
Optional.  
An ISO 3166-1 alpha-2 country code or the string "from_token".  
If a country code is specified, only artists with content that is playable in that market is returned.  
Note:  
- If market is set to "from_token", and a valid access token is specified in the request header, only content playable in the country associated with the user account is returned.  
- Users can view the country that is associated with their account in the account settings. A user must grant access to the user-read-private scope prior to when the access token is issued.

limit  
Optional.  
Maximum number of results to return.  
Default: 20  
Minimum: 1  
Maximum: 50

offset  
Optional.  
The index of the first result to return.  
Default: 0 (the first result).  
Maximum offset (including limit): 10,000.  
Use with limit to get the next page of search results.

authorization  
Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to spotifyr::get_spotify_access_token()

Examples

## Not run:
get_genre_artists('wonky')

## End(Not run)
get_label_artists  

Search for artists by label

Description
Get Spotify Catalog information about artists belonging to a given label.

Usage
get_label_artists(label = character(), market = NULL, limit = 20, offset = 0, authorization = get_spotify_access_token())

Arguments
- **label**  
  Required.
  String of label name to search for
  For example: label = "brainfeeder".
- **market**  
  Optional.
  An ISO 3166-1 alpha-2 country code or the string "from_token".
  If a country code is specified, only artists with content that is playable in that market is returned.
  Note:
  - If market is set to "from_token", and a valid access token is specified in the request header, only content playable in the country associated with the user account is returned.
  - Users can view the country that is associated with their account in the account settings. A user must grant access to the user-read-private scope prior to when the access token is issued.
- **limit**  
  Optional.
  Maximum number of results to return.
  Default: 20
  Minimum: 1
  Maximum: 50
- **offset**  
  Optional.
  The index of the first result to return.
  Default: 0 (the first result).
  Maximum offset (including limit): 10,000.
  Use with limit to get the next page of search results.
- **authorization**  
  Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to spotifyr::get_spotify_access_token()

Examples
```r
## Not run:
get_label_artists('brainfeeder')

## End(Not run)
```
get_my_currently_playing

Get the object currently being played on the user’s Spotify account.

Description

Get the object currently being played on the user’s Spotify account.

Usage

```r
get_my_currently_playing(market = NULL,
authorization = get_spotify_authorization_code())
```

Arguments

- **market**
  - An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking

- **authorization**
  - Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the user-read-currently-playing and/or user-read-playback-state scope authorized in order to read information.

Value


get_my_current_playback

Get information about the user’s current playback state, including track, track progress, and active device.

Description

Get information about the user’s current playback state, including track, track progress, and active device.

Usage

```r
get_my_current_playback(market = NULL,
authorization = get_spotify_authorization_code())
```
get_my_devices

**Arguments**

- **market**
  - An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking.

- **authorization**
  - Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the `user-read-playback-state` scope authorized in order to read information.

**Value**

Returns a list containing user playback information. See the official Spotify Web API documentation for more information.

---

**get_my_devices**

*Get information about a user’s available devices.*

**Description**

Get information about a user's available devices.

**Usage**

```
get_my_devices(authorization = get_spotify_authorization_code())
```

**Arguments**

- **authorization**
  - Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the `user-read-playback-state` scope authorized in order to read information.

**Value**

Returns a data frame of results containing user device information. See the official Spotify Web API documentation for more information.
get_my_followed_artists

*Get the current user’s followed artists.*

**Description**

Get the current user’s followed artists.

**Usage**

```
get_my_followed_artists(limit = 20, after = NULL,
authorization = get_spotify_authorization_code(),
include_meta_info = FALSE)
```

**Arguments**

- **limit**
- **after**
  Optional. The last artist ID retrieved from the previous request.
- **authorization**
  Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`. The access token must have been issued on behalf of the current user. Getting details of the artists or users the current user follows requires authorization of the user-follow-read scope. See Using Scopes.
- **include_meta_info**
  Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

get_my_playlists

*Get a list of the playlists owned or followed by the current Spotify user.*

**Description**

Get a list of the playlists owned or followed by the current Spotify user.

**Usage**

```
get_my_playlists(limit = 20, offset = 0,
authorization = get_spotify_authorization_code(),
include_meta_info = FALSE)
```
Arguments

limit  
Optional.
Maximum number of playlists to return.
Default: 20
Minimum: 1
Maximum: 50

offset  
Optional.
The index of the first playlist to return.
Default: 0 (the first object). Maximum offset: 100,000. Use with limit to get the next set of playlists.

authorization  
Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`.
The access token must have been issued on behalf of the current user.
Private playlists are only retrievable for the current user and requires the `playlist-read-private` scope to have been authorized by the user. Note that this scope alone will not return collaborative playlists, even though they are always private.
Collaborative playlists are only retrievable for the current user and requires the `playlist-read-collaborative` scope to have been authorized by the user.

include_meta_info  
Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

Value


code

```
get_my_profile
```

Get detailed profile information about the current user (including the current user’s username).

Description

Get detailed profile information about the current user (including the current user’s username).

Usage

```
get_my_profile(authorization = get_spotify_authorization_code())
```
get_my_recently_played

Arguments

authorization  Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`. The access token must have been issued on behalf of the current user. Reading the user’s email address requires the `user-read-email` scope; reading country and product subscription level requires the `user-read-private` scope. Reading the user’s birthdate requires the `user-read-birthdate` scope. See Using Scopes.

Value


get_my_recently_played

Get Current User’s Recently Played Tracks

Description

Get Current User’s Recently Played Tracks

Usage

```r
get_my_recently_played(limit = 20, after = NULL, before = NULL,
                        authorization = get_spotify_authorization_code(),
                        include_meta_info = FALSE)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>after</td>
<td>Optional. A Unix timestamp in milliseconds. Returns all items after (but not including) this cursor position. If after is specified, before must not be specified.</td>
</tr>
<tr>
<td>before</td>
<td>Optional. A Unix timestamp in milliseconds. Returns all items before (but not including) this cursor position. If before is specified, after must not be specified.</td>
</tr>
<tr>
<td>authorization</td>
<td>Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to <code>spotifyr::get_spotify_access_token()</code>. The access token must have been issued on behalf of the current user.</td>
</tr>
<tr>
<td>include_meta_info</td>
<td>Optional. Boolean indicating whether to include full result, with meta information such as &quot;before&quot;, &quot;after&quot;, and &quot;limit&quot;. Defaults to FALSE. The access token must have the user-read-recently-played scope authorized in order to read the user’s recently played tracks.</td>
</tr>
</tbody>
</table>
**get_my_top_artists_or_tracks**

*Get the current user’s top artists or tracks based on calculated affinity.*

**Description**

Get the current user’s top artists or tracks based on calculated affinity.

**Usage**

```r
get_my_top_artists_or_tracks(type = NULL, limit = 20, offset = 0,
time_range = "medium_term",
authorization = get_spotify_authorization_code(),
include_meta_info = FALSE)
```

**Arguments**

- `type`: Required. The type of entity to return. Valid values: `artists` or `tracks`.
- `limit`: Optional. Maximum number of results to return. Default: 20
  - Minimum: 1
  - Maximum: 50
- `offset`: Optional. The index of the first entity to return. Default: 0 (i.e., the first track). Use with `limit` to get the next set of entities.
- `time_range`: Optional. Over what time frame the affinities are computed. Valid values: `long_term` (calculated from several years of data and including all new data as it becomes available), `medium_term` (approximately last 6 months), `short_term` (approximately last 4 weeks). Default: `medium_term`.
- `authorization`: Required. A valid access token from the Spotify Accounts service. See the [Web API authorization Guide](#) for more details. Defaults to `spotifyr::get_spotify_access_token()`.
- `include_meta_info`: Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to `FALSE`.

**Value**

Returns a data frame of results containing track or album data. See the official API [documentation](#) for more information.
get_new_releases

Get a list of new album releases featured in Spotify (shown, for example, on a Spotify player’s “Browse” tab).

Description

Get a list of new album releases featured in Spotify (shown, for example, on a Spotify player’s “Browse” tab).

Usage

get_new_releases(country = NULL, limit = 20, offset = 0,
authorization = get_spotify_access_token(),
include_meta_info = FALSE)

Arguments

country  Optional. A country: an ISO 3166-1 alpha-2 country code. Provide this parameter if you want the list of returned items to be relevant to a particular country. If omitted, the returned items will be relevant to all countries.


offset  Optional. The index of the first item to return. Default: 0 (the first object). Use with limit to get the next set of items.

authorization  Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to spotifyr::get_spotify_access_token()

include_meta_info  Optional. Boolean indicating whether to include full result, with meta information such as "country", "offset", and "limit". Defaults to FALSE.

Value


Examples

```r
## Not run:
## Get new Swedish music
get_new_releases(country = 'SE')

## End(Not run)
```
get_playlist

Get a playlist owned by a Spotify user.

Description

Get a playlist owned by a Spotify user.

Usage

get_playlist(playlist_id, fields = NULL, market = NULL,
authorization = get_spotify_access_token())

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>playlist_id</td>
<td>Required. The Spotify ID for the playlist.</td>
</tr>
</tbody>
</table>
| fields       | Optional. Filters for the query: a comma-separated list of the fields to return. If omitted, all fields are returned. For example, to get just the playlist's description and URI: fields = c("description", "uri") A dot separator can be used to specify non-reoccurring fields, while parentheses can be used to specify reoccurring fields within objects. For example, to get just the added date and user ID of the adder: fields = "tracks.items(added_at,added_by.id)" Use multiple parentheses to drill down into nested objects, for example: fields = "tracks.items(track(name,href,album(name,href)))" Fields can be excluded by prefixing them with an exclamation mark, for example: fields = "tracks.items(track(name,href,album(!name,href)))"
| market       | Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking |
| authorization| Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Both Public and Private playlists belonging to any user are retrievable on provision of a valid access token. Defaults to spotifyr::get_spotify_access_token() |

Value

get_playlist_audio_features

Get features and popularity for all of a given set of playlists on Spotify

Description

This function returns the popularity and audio features for every song for a given set of playlists on Spotify.

Usage

get_playlist_audio_features(username, playlist_uris, authorization = get_spotify_access_token())

Arguments

username String of Spotify username. Can be found on the Spotify app.
playlist_uris Character vector of Spotify playlist uris. Can be found within the Spotify App.
authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to spotifyr::get_spotify_access_token()

Examples

## Not run:
playlist_username <- 'spotify'
playlist_uris <- c('37i9dQZF1E9T1oFsQFGg9K', '37i9dQZF1CyQNO1T1QVf3p')
playlist_audio_features <- get_playlist_audio_features(playlist_username, playlist_uris)

## End(Not run)

get_playlist_cover_image

Get the current image associated with a specific playlist.

Description

Get the current image associated with a specific playlist.

Usage

get_playlist_cover_image(playlist_id, authorization = get_spotify_authorization_code())
get_playlist_tracks

Arguments

playlist_id Required. The Spotify ID for the playlist.
authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_authorization_code()`. The access token must have been issued on behalf of the current user. Current playlist image for both Public and Private playlists of any user are retrievable on provision of a valid access token.

Value

Returns a data frame of results containing playlist cover image information. See the official Spotify Web API Documentation for more information.

get_playlist_tracks  Get full details of the tracks of a playlist owned by a Spotify user.

Description

Get full details of the tracks of a playlist owned by a Spotify user.

Usage

```r
get_playlist_tracks(playlist_id, fields = NULL, limit = 100,
offset = 0, market = NULL,
authorization = get_spotify_access_token(),
include_meta_info = FALSE)
```

Arguments

playlist_id Required. The Spotify ID for the playlist.
fields Optional. Filters for the query: a comma-separated list of the fields to return. If omitted, all fields are returned. For example, to get just the playlist's description and URI: `fields = c("description", "uri")`. A dot separator can be used to specify non-reoccurring fields, while parentheses can be used to specify re-occurring fields within objects. For example, to get just the added date and user ID of the adder:

```r
fields = "tracks.items(added_at,added_by.id)". Use multiple parentheses to drill down into nested objects, for example:
```

```r
fields = "tracks.items(track(name,href,album(name,href)))". Fields can be excluded by prefixing them with an exclamation mark, for example:
```

```r
fields = "tracks.items(track(name,href,album(!name,href)))"
```

limit Optional. Maximum number of tracks to return. Default: 100 Minimum: 1 Maximum: 100
get_recommendations

Offset

Optional.
The index of the first track to return.
Default: 0 (the first object).

Market

Optional.
An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking

Authorization

Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Both Public and Private playlists belonging to any user are retrievable on provision of a valid access token. Defaults to

Include meta info

Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

Value


get_recommendations

Create a playlist-style listening experience based on seed artists, tracks and genres.

Description

Create a playlist-style listening experience based on seed artists, tracks and genres.

Usage

```
get_recommendations(limit = 20, market = NULL, seed_artists = NULL, seed_genres = NULL, seed_tracks = NULL, max_acousticness = NULL, max_danceability = NULL, max_duration_ms = NULL, max_energy = NULL, max_instrumentalness = NULL, max_key = NULL, max_liveness = NULL, max_loudness = NULL, max_mode = NULL, max_popularity = NULL, max_speechiness = NULL, max_tempo = NULL, max_time_signature = NULL, max_valence = NULL, min_acousticness = NULL, min_danceability = NULL, min_duration_ms = NULL, min_energy = NULL, min_instrumentalness = NULL, min_key = NULL, min_liveness = NULL, min_loudness = NULL, min_mode = NULL, min_popularity = NULL, min_speechiness = NULL, min_tempo = NULL, min_time_signature = NULL, min_valence = NULL, target_acousticness = NULL, target_danceability = NULL, target_duration_ms = NULL, target_energy = NULL,)
```
target_instrumentalness = NULL, target_key = NULL, target_liveness = NULL, target_loudness = NULL, target_mode = NULL, target_popularity = NULL, target_speechiness = NULL, target_tempo = NULL, target_time_signature = NULL, target_valence = NULL, authorization = get_spotify_access_token(), include_seeds_in_response = FALSE)

Arguments

limit   Optional. The target size of the list of recommended tracks. For seeds with unusually small pools or when highly restrictive filtering is applied, it may be impossible to generate the requested number of recommended tracks. Debugging information for such cases is available in the response. Default: 20. Minimum: 1. Maximum: 100.

market  Optional. An ISO 3166-1 alpha-2 country code or the string from_token. Provide this parameter if you want to apply Track Relinking. Because min_* , max_* and target_* are applied to pools before relinking, the generated results may not precisely match the filters applied. Original, non-relinked tracks are available via the linked_from attribute of the relinked track response.

seed_artists A character vector of Spotify IDs for seed artists. Up to 5 seed values may be provided in any combination of seed_artists, seed_tracks and seed_genres.

seed_genres A character vector of any genres in the set of available genre seeds. Up to 5 seed values may be provided in any combination of seed_artists, seed_tracks and seed_genres.

seed_tracks A character vector of Spotify IDs for a seed track. Up to 5 seed values may be provided in any combination of seed_artists, seed_tracks and seed_genres.

max_acousticness Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ acousticness.

max_danceability Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ danceability.

max_duration_ms Optional. Integer indicating a hard ceiling on recommended tracks’ duration in milliseconds.

max_energy Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ energy.

max_instrumentalness Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ instrumentalness.

max_key Optional. Integer ranging from 1 to 12 indicating a hard ceiling on recommended tracks’ key.

max_liveness Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ liveness.

max_loudness Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ loudness.
get_recommendations

max_mode          Optional. Integer ranging from 0 to 1 indicating a hard ceiling on recommended tracks’ mode.
max_popularity    Optional. Integer ranging between 0 and 100 indicating a hard ceiling on recommended tracks’ popularity.
max_speechiness   Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ speechiness.
max_tempo         Optional. Numeric value indicating a hard ceiling on recommended tracks’ tempo.
max_time_signature Optional. Integer indicating a hard ceiling on recommended tracks’ time signature.
max_valence       Optional. Numeric value between 0 and 1 indicating a hard ceiling on recommended tracks’ valence.
min_acousticness  Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ acousticness.
min_danceability  Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ danceability.
min_duration_ms   Optional. Integer indicating a hard floor on recommended tracks’ duration in milliseconds.
min_energy        Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ energy.
min_instrumentalness Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ instrumentalness.
min_key           Optional. Integer ranging from 1 to 12 indicating a hard floor on recommended tracks’ key.
min_liveness      Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ liveness.
min_loudness      Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ loudness.
min_mode          Optional. Integer ranging from 0 to 1 indicating a hard floor on recommended tracks’ mode.
min_popularity    Optional. Integer ranging between 0 and 100 indicating a hard floor on recommended tracks’ popularity.
min_speechiness   Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks’ speechiness.
min_tempo         Optional. Numeric value indicating a hard floor on recommended tracks’ tempo.
min_time_signature Optional. Integer indicating a hard floor on recommended tracks’ time signature.
get_recommendations

min_valence  Optional. Numeric value between 0 and 1 indicating a hard floor on recommended tracks' valence.

target_acousticness  Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks' acousticness. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request target_energy = 0.6 and target_danceability = 0.8. All target values will be weighed equally in ranking results.

target_danceability  Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks' danceability. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request target_energy = 0.6 and target_danceability = 0.8. All target values will be weighed equally in ranking results.

target_duration_ms  Optional. Integer indicating a target value for recommended tracks' duration in milliseconds. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request target_energy = 0.6 and target_danceability = 0.8. All target values will be weighed equally in ranking results.

target_energy  Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks' energy. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request target_energy = 0.6 and target_danceability = 0.8. All target values will be weighed equally in ranking results.

target_instrumentalness  Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks' instrumentalness. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request target_energy = 0.6 and target_danceability = 0.8. All target values will be weighed equally in ranking results.

target_key  Optional. Integer ranging from 1 to 12 indicating a target value for recommended tracks' key. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request target_energy = 0.6 and target_danceability = 0.8. All target values will be weighed equally in ranking results.

target_liveness  Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks' liveness. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request target_energy = 0.6 and target_danceability = 0.8. All target values will be weighed equally in ranking results.

target_loudness  Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks' loudness. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request target_energy = 0.6 and target_danceability = 0.8. All target values will be weighed equally in ranking results.
get_recommendations

**target_mode**  
Optional. Integer ranging from 0 to 1 indicating a target value for recommended tracks’ mode. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6` and `target_danceability = 0.8`. All target values will be weighed equally in ranking results.

**target_popularity**  
Optional. Integer ranging between 0 and 100 indicating a target value for recommended tracks’ popularity. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6` and `target_danceability = 0.8`. All target values will be weighed equally in ranking results.

**target_speechiness**  
Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks’ speechiness. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6` and `target_danceability = 0.8`. All target values will be weighed equally in ranking results.

**target_tempo**  
Optional. Numeric value indicating a target value for recommended tracks’ tempo. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6` and `target_danceability = 0.8`. All target values will be weighed equally in ranking results.

**target_time_signature**  
Optional. Integer indicating a target value for recommended tracks’ time signature. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6` and `target_danceability = 0.8`. All target values will be weighed equally in ranking results.

**target_valence**  
Optional. Numeric value between 0 and 1 indicating a target value for recommended tracks’ valence. Tracks with the attribute values nearest to the target values will be preferred. For example, you might request `target_energy = 0.6` and `target_danceability = 0.8`. All target values will be weighed equally in ranking results.

**authorization**  
Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to `spotifyr::get_spotify_access_token()`.

**include_seeds_in_response**  
Optional. Boolean for whether to include seed object in response. Defaults to `FALSE`.

**Value**

Returns a data frame of results recommendations. See the official Spotify Web API documentation for more information.

**Examples**

```
## Not run:
## Get new Swedish music
get_recommendations(country = 'SE')
```
get_spotify_access_token

### Description

This function creates a Spotify access token.

### Usage

```r
get_spotify_access_token(client_id = Sys.getenv("SPOTIFY_CLIENT_ID"),
client_secret = Sys.getenv("SPOTIFY_CLIENT_SECRET"))
```

get_related_artists

Get Spotify catalog information about artists similar to a given artist. Similarity is based on analysis of the Spotify community’s listening history.

### Description

Get Spotify catalog information about artists similar to a given artist. Similarity is based on analysis of the Spotify community’s listening history.

### Usage

```r
get_related_artists(id, authorization = get_spotify_access_token(),
include_meta_info = FALSE)
```

### Arguments

- **id**: The Spotify ID for the artist.
- **authorization**: Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to `spotifyr::get_spotify_access_token()`.
- **include_meta_info**: Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to `FALSE`.

### Value

get_spotify_authorization_code

Arguments

client_id Defaults to System Environment variable "SPOTIFY_CLIENT_ID"
client_secret Defaults to System Environment variable "SPOTIFY_CLIENT_SECRET"

Examples

```r
## Not run:
get_spotify_access_token()

## End(Not run)
```

### Description

This function creates a Spotify access token.

### Usage

```r
get_spotify_authorization_code(client_id = Sys.getenv("SPOTIFY_CLIENT_ID"),
client_secret = Sys.getenv("SPOTIFY_CLIENT_SECRET"),
scope = spotyrf::scopes)
```

### Arguments

- **client_id**: Defaults to System Environment variable "SPOTIFY_CLIENT_ID"
- **client_secret**: Defaults to System Environment variable "SPOTIFY_CLIENT_SECRET"
- **scope**: Space delimited string of spotify scopes, found here: https://developer.spotify.com/documentation/general/guides/scopes/. All scopes are selected by default

### Examples

```r
## Not run:
get_spotify_authorization_code()

## End(Not run)
```
get_track

Get Spotify catalog information for a single track identified by its unique Spotify ID.

Description
Get Spotify catalog information for a single track identified by its unique Spotify ID.

Usage
get_track(id, market = NULL, authorization = get_spotify_access_token())

Arguments
- id: The Spotify ID for the track.
- market: Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking.
- authorization: Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to `spotifyr::get_spotify_access_token()`.

Value
Returns a data frame of results containing track data. See https://developer.spotify.com/documentation/web-api/reference/tracks/get-several-tracks/ for more information.

get_tracks

Get Spotify catalog information for a single track identified by its unique Spotify ID.

Description
Get Spotify catalog information for a single track identified by its unique Spotify ID.

Usage
get_tracks(ids, market = NULL, authorization = get_spotify_access_token(), include_meta_info = FALSE)
get_track_audio_analysis

Arguments

- ids: The Spotify ID for the track.
- market: Optional. An ISO 3166-1 alpha-2 country code or the string "from_token". Provide this parameter if you want to apply Track Relinking.
- authorization: Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to `spotifyr::get_spotify_access_token()`.
- include_meta_info: Optional. Boolean indicating whether to include result, with meta information such as "total", and "limit". Defaults to FALSE.

Value

Returns a data frame of results containing track audio analysis data. See https://developer.spotify.com/documentation/web-api/reference/tracks/get-audio-analysis/ for more information.

Description

Get a detailed audio analysis for a single track identified by its unique Spotify ID.

Usage

```r
get_track_audio_analysis(id, authorization = get_spotify_access_token())
```

Arguments

- id: The Spotify ID for the track.
- authorization: Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to `spotifyr::get_spotify_access_token()`.

Value

Returns a data frame of results containing track audio analysis data. See https://developer.spotify.com/documentation/web-api/reference/tracks/get-audio-analysis/ for more information.
get_track_audio_features

Get audio feature information for a single track identified by its unique Spotify ID.

Description

Get audio feature information for a single track identified by its unique Spotify ID.

Usage

get_track_audio_features(ids, authorization = get_spotify_access_token())

Arguments

- ids: Required. A comma-separated list of the Spotify IDs of the tracks. Maximum: 100 IDs.
- authorization: Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to spotifyr::get_spotify_access_token()

Value

Returns a data frame of results containing track audio features data. See https://developer.spotify.com/documentation/web-api/reference/tracks/get-several-audio-features/ for more information.

get_user_audio_features

Get audio feature information for a users' playlists

Description

Get audio feature information for a users' playlists

Usage

get_user_audio_features(username = NULL, authorization = get_spotify_access_token())

Arguments

- username: Required. String of Spotify username.
- authorization: Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to spotifyr::get_spotify_access_token()
get_user_playlists

Value

Returns a data frame of results containing track audio features data. See the Spotify Web API documentation for more information.

Description

Get a list of the playlists owned or followed by a Spotify user.

Usage

get_user_playlists(user_id, limit = 20, offset = 0,
                   authorization = get_spotify_authorization_code(),
                   include_meta_info = FALSE)

Arguments

user_id Required. The user’s Spotify user ID.
limit Optional. Maximum number of playlists to return. Default: 20

offset Optional. The index of the first playlist to return. Default: 0 (the first object). Maximum offset: 100,000. Use with limit to get the next set of playlists.
authorization Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to spotifyr::get_spotify_authorization_code().

include_meta_info Optional. Boolean indicating whether to include full result, with meta information such as "total", and "limit". Defaults to FALSE.

Value

Returns a data frame of results containing user playlist information. See the official Spotify Web API documentation for more information.
**get_user_profile**  
*Get public profile information about a Spotify user.*

**Description**

Get public profile information about a Spotify user.

**Usage**

```R
get_user_profile(user_id, authorization = get_spotify_access_token())
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user_id</td>
<td>Required. The user’s Spotify user ID.</td>
</tr>
<tr>
<td>authorization</td>
<td>Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to <code>spotifyr::get_spotify_access_token()</code></td>
</tr>
</tbody>
</table>

**Value**


---

**is_uri**  
*Check if a string matches the pattern of a Spotify URI*

**Description**

Check if a string matches the pattern of a Spotify URI

**Usage**

```R
is_uri(s)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>String to check</td>
</tr>
</tbody>
</table>
pause_my_playback    

Pause playback on the user’s account.

Description

Pause playback on the user’s account.

Usage

```python
pause_my_playback(device_id = NULL, 
                   authorization = get_spotify_authorization_code())
```

Arguments

device_id    Optional. The id of the device this command is targeting. If not supplied, the user’s currently active device is the target.

authorization    Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the `user-modify-playback-state` scope authorized in order to control playback.

pitch_class_lookup    

Pitch class notation lookup

Description

Pitch class notation lookup

Usage

```python
pitch_class_lookup
```

Format

An object of class `character` of length 12.
 scopes | Valid scopes
---|---

**Description**

Vector of valid scopes for spotifyr::get_authorization_code()

**Usage**

`scopes`

**Format**

An object of class character of length 17.

**search_spotify**

**Search for an item**

**Description**

Get Spotify Catalog information about artists, albums, tracks or playlists that match a keyword string. For more information see the official documentation.

**Usage**

```r
search_spotify(q, type = c("album", "artist", "playlist", "track"),
             market = NULL, limit = 20, offset = 0, include_external = NULL,
             authorization = get_spotify_access_token())
```

**Arguments**

- **q**
  
  Required.
  
  Search query keywords and optional field filters and operators.

- **type**
  
  A character vector of item types to search across.
  
  Valid types are album, artist, playlist, and track.
  
  Search results include hits from all the specified item types.
  
  For example: `q = "name:abacab" and type =c("album", "track")` returns both albums and tracks with "abacab" included in their name.

- **market**
  
  Optional.
  
  An ISO 3166-1 alpha-2 country code or the string "from_token".
  
  If a country code is specified, only artists, albums, and tracks with content that is playable in that market is returned.
  
  Note:
  
  - Playlist results are not affected by the market parameter.
  - If market is set to "from_token", and a valid access token is specified in the
seek_to_position

request header, only content playable in the country associated with the user account, is returned.
- Users can view the country that is associated with their account in the account settings. A user must grant access to the user-read-private scope prior to when the access token is issued.

**limit**
- Optional.
- Maximum number of results to return.
- Default: 20
- Minimum: 1
- Maximum: 50
- Note: The limit is applied within each type, not on the total response.
- For example, if the limit value is 3 and the type is "artist", "album", the response contains 3 artists and 3 albums.

**offset**
- Optional.
- The index of the first result to return.
- Default: 0 (the first result).
- Maximum offset (including limit): 10,000.
- Use with limit to get the next page of search results.

**include_external**
- Optional.
- Possible values: audio
- If include_external = "audio" is specified the response will include any relevant audio content that is hosted externally.
- By default external content is filtered out from responses.

**authorization**
- Required. A valid access token from the Spotify Accounts service. See the Web API authorization guide for more details. Defaults to `spotifyr::get_spotify_access_token()`

**Examples**

```r
## Not run:
search_spotify('radiohead', 'artist')

## End(Not run)
```

---

**seek_to_position**

Seeks to the given position in the user’s currently playing track.

**Description**

Seeks to the given position in the user’s currently playing track.

**Usage**

```r
seek_to_position(position_ms, device_id = NULL, 
authorization = get_spotify_authorization_code())
```
set_my_repeat_mode

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>position_ms</td>
<td>Required. Integer indicating the position in milliseconds to seek to. Must be a positive number. Passing in a position that is greater than the length of the track will cause the player to start playing the next song.</td>
</tr>
<tr>
<td>device_id</td>
<td>Optional. The id of the device this command is targeting. If not supplied, the user’s currently active device is the target.</td>
</tr>
<tr>
<td>authorization</td>
<td>Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to <code>spotifyr::get_spotify_access_token()</code>. The access token must have been issued on behalf of the current user. The access token must have the <code>user-modify-playback-state</code> scope authorized in order to control playback.</td>
</tr>
</tbody>
</table>

**set_my_repeat_mode**

Set the repeat mode for the user’s playback. Options are repeat-track, repeat-context, and off.

Description

Set the repeat mode for the user’s playback. Options are repeat-track, repeat-context, and off.

Usage

```cpp
set_my_repeat_mode(state, device_id = NULL,
                   authorization = get_spotify_authorization_code())
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>Required.</td>
</tr>
<tr>
<td>device_id</td>
<td>Optional. The id of the device this command is targeting. If not supplied, the user’s currently active device is the target.</td>
</tr>
<tr>
<td>authorization</td>
<td>Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to <code>spotifyr::get_spotify_access_token()</code>. The access token must have been issued on behalf of the current user. The access token must have the <code>user-modify-playback-state</code> scope authorized in order to control playback.</td>
</tr>
</tbody>
</table>
set_my_volume

**Set the volume for the user's current playback device.**

**Description**

Set the volume for the user's current playback device.

**Usage**

```python
set_my_volume(volume_percent, device_id = NULL,
               authorization = get_spotify_authorization_code())
```

**Arguments**

- **volume_percent** Required. Integer. The volume to set. Must be a value from 0 to 100 inclusive.
- **device_id** Optional. The id of the device this command is targeting. If not supplied, the user's currently active device is the target.
- **authorization** Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the `user-modify-playback-state` scope authorized in order to control playback.

**skip_my_playback**

**Skips to next track in the user's queue.**

**Description**

Skips to next track in the user’s queue.

**Usage**

```python
skip_my_playback(device_id = NULL,
                  authorization = get_spotify_authorization_code())
```

**Arguments**

- **device_id** Optional. The id of the device this command is targeting. If not supplied, the user’s currently active device is the target.
- **authorization** Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the `user-modify-playback-state` scope authorized in order to control playback.
skip_my_playback_previous

Skips to previous track in the user’s queue.

Description

Skips to previous track in the user’s queue.

Usage

```python
skip_my_playback_previous(device_id = NULL,
authorization = get_spotify_authorization_code())
```

Arguments

device_id  
Optional. The id of the device this command is targeting. If not supplied, the user’s currently active device is the target.

authorization  
Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the `user-modify-playback-state` scope authorized in order to control playback.

spotifyr  

Description

A Quick and Easy Wrapper for Pulling Track Audio Features from Spotify’s Web API in Bulk

Details

See the README on GitHub
**start_my_playback**

**Skips to previous track in the user's queue.**

### Description

Skips to previous track in the user's queue.

### Usage

```r
start_my_playback(device_id = NULL, context_uri = NULL, uris = NULL,
                   offset = NULL, position_ms = NULL,
                   authorization = get_spotify_authorization_code())
```

### Arguments

- **device_id**
  - Optional. The id of the device this command is targeting. If not supplied, the user's currently active device is the target.

- **context_uri**
  - Optional. String of the Spotify URI of the context to play. Valid contexts are albums, artists, playlists. Example `context_uri = "spotify:album:1Je1IMUbXcx1Fz0WE7oPT"`.

- **uris**
  - Optional. A character vector of the Spotify track URIs to play. For example:
    ```r
    uris <- c("spotify:track:4iV5W9uYEEdYUVa79Axb7Rh", "spotify:track:1301W1eyT98SxVHPZCj")
    ```

- **offset**
  - Optional. A named list indicating from where the context playback should start. Only available when `context_uri` corresponds to an album or playlist object, or when the `uris` parameter is used.
  - 
    ```r
    offset <- list(position = 5)
    ```

- **position_ms**
  - Optional. Integer indicating from what position to start playback. Must be a positive number. Passing in a position that is greater than the length of the track will cause the player to start playing the next song.

- **authorization**
  - Required. A valid access token from the Spotify Accounts service. See the Web API [authorization Guide](https://developer.spotify.com/documentation/web-api/guides/authorization-guide/) for more details. Defaults to `spotifyr::get_spotify_access_token()`. The access token must have been issued on behalf of the current user. The access token must have the `user-modify-playback-state` scope authorized in order to control playback.

**toggle_my_shuffle**

**Toggle shuffle on or off for user's playback.**

### Description

Toggle shuffle on or off for user's playback.
Usage

```r
toggle_my_shuffle(state, device_id = NULL,
    authorization = get_spotify_authorization_code())
```

Arguments

- **state**: Required. 
  - TRUE: Shuffle user’s playback 
  - FALSE: Do not shuffle user’s playback 

- **device_id**: Optional. The id of the device this command is targeting. If not supplied, the user’s currently active device is the target. 

- **authorization**: Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. 
  - The access token must have been issued on behalf of the current user. 
  - The access token must have the `user-modify-playback-state` scope authorized in order to control playback.

---

**transfer_my_playback**

Transfer playback to a new device and determine if it should start playing.

Usage

```r
transfer_my_playback(device_ids, play = FALSE,
    authorization = get_spotify_authorization_code())
```

Arguments

- **device_ids**: Required. A character vector containing the ID of the device this on which playback should be started/transferred. Note: only a single device_id is currently supported. 

- **play**: Optional. 
  - TRUE: Ensure playback happens on new device 
  - FALSE (default): keep the current playback state 
  
  Note that a value of FALSE for the `play` parameter when also transferring to another device_id will not pause playback. To ensure that playback is paused on the new device you should send a pause command to the currently active device before transferring to the new device_id. 

- **authorization**: Required. A valid access token from the Spotify Accounts service. See the Web API authorization Guide for more details. Defaults to `spotifyr::get_spotify_access_token()`. 
  - The access token must have been issued on behalf of the current user. 
  - The access token must have the `user-modify-playback-state` scope authorized in order to control playback.
**unfollow_playlist**  
Remove the current user as a follower of a playlist.

**Description**  
Remove the current user as a follower of a playlist.

**Usage**  
```
unfollow_playlist(playlist_id,
    authorization = get_spotify_authorization_code())
```

**Arguments**  
- **playlist_id**  
  Required. The Spotify ID of the playlist that is to be no longer followed.
- **authorization**  
  Required. A valid access token from the Spotify Accounts service. See the [Web API authorization Guide](#) for more details. Defaults to `spotifyr::get_spotify_authorization_code()`. The access token must have been issued on behalf of the current user. Unfollowing a publicly followed playlist for a user requires authorization of the `playlist-modify-public` scope; unfollowing a privately followed playlist requires the `playlist-modify-private` scope. See [Using Scopes](#). Note that the scopes you provide relate only to whether the current user is following the playlist publicly or privately (i.e. showing others what they are following), not whether the playlist itself is public or private.

---

**verify_result**  
Verify API result

**Description**  
Check API result for error codes

**Usage**  
```
verify_result(res)
```

**Arguments**  
- **res**  
  API result ot check
Index

*Topic **artist**
  get_genre_artists, 19
  get_label_artists, 20

*Topic **audio**
  get_playlist_audio_features, 29

*Topic **auth**
  get_spotify_access_token, 36
  get_spotify_authorization_code, 37

*Topic **datasets**
  pitch_class_lookup, 43
  scopes, 44

*Topic **features**
  get_playlist_audio_features, 29

*Topic **label**
  get_genre_artists, 19
  get_label_artists, 20

*Topic **playlists**
  get_playlist_audio_features, 29

*Topic **search**
  get_genre_artists, 19
  get_label_artists, 20
  search_spotify, 44

*Topic **track**
  get_playlist_audio_features, 29

add_tracks_to_playlist, 3
change_playlist_details, 4
check_me_following, 4
check_users_following, 5
create_playlist, 6

dedupe_album_names, 6

follow_artists_or_users, 7
follow_playlist, 8

get_album, 8
get_album_data, 10
get_album_tracks, 10
get_albums, 9

generate_playlist_audio_features, 29

get_artist, 11
get_artist_albums, 12
get_artist_audio_features, 13
get_artist_top_tracks, 14
get_artists, 12
get_category, 15
get_category_playlists, 16
get_discography, 17
get_featured_playlists, 17
get_genre_artists, 19
get_label_artists, 20
get_my_current_playback, 21
get_my_currently_playing, 21
get_my_devices, 22
get_my_followed_artists, 23
get_my_playlists, 23
get_my_profile, 24
get_my_recently_played, 25
get_my_top_artists_or_tracks, 26
get_new_releases, 27
get_playlist, 28
get_playlist_audio_features, 29
get_playlist_cover_image, 29
get_playlist_tracks, 30
get_recommendations, 31
get_related_artists, 36
get_spotify_access_token, 36
get_spotify_authorization_code, 37
get_track, 38
get_track_audio_analysis, 39
get_track_audio_features, 40
get_tracks, 38
get_user_audio_features, 40
get_user_playlists, 41
get_user_profile, 42
is_uri, 42
pause_my_playback, 43
pitch_class_lookup, 43
scopes, 44
search_spotify, 44
seek_to_position, 45
set_my_repeat_mode, 46
set_my_volume, 47
skip_my_playback, 47
skip_my_playback_previous, 48
spotifir, 48
spotifir-package (spotifir), 48
start_my_playback, 49

toggle_my_shuffle, 49
transfer_my_playback, 50

unfollow_playlist, 51

verify_result, 51